AD-A031 459 GENERAL RESEARCH CORP SANTA BARBARA CALIF F/G 9/2
STRUCTURED PROGRAMMING TRANSLATORS. VOLUME III. STRUCTRAN-1 SYS--ETC(U)
AUG 76 D M ANDREWS, R A MELTON F30602-75-C-0245

UNCLASSIFIED RADC-TR-76-253-VOL-3 NL

1 OF 1
AD
A031459

END

DATE
FILMED
12-76

RADC-TR-76-253, Vol III (of five)
Final Technical Report
August 1976

STRUCTURED PROGRAMMING TRANSLATORS

STRUCTRAN-1 System Design and Implementation Manual

General Research Corporation

Approved for public release;
distribution unlimited.

ROME AIR DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
GRIFFISS AIR FORCE BASE, NEW YORK 13441

This report consists of the following volumes:

I - Final Report
II - STRUCTRAN-1 User's Manual
III - STRUCTRAN-1 System Design and Implementation Manual
IV - STRUCTRAN-2 User's Manual
V - STRUCTRAN-2 System Design and Implementation Manual

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

This report has been reviewed and approved for publication.

APPROVED: Donald L. Mark

DONALD L. MARK
Project Engineer

APPROVED: Robert D. Krutz

ROBERT D. KRUTZ, Col, USAF
Chief, Information Sciences Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.

AMERICAN REVOLUTION BICENTENNIAL
1776-1976

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS
BEFORE COMPLETING FORM

18 1. REPORT NUMBER
RADC-TR-76-253-Vol-3

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER
9

6 4. TITLE (and Subtitle)
STRUCTURED PROGRAMMING TRANSLATORS. Volume III. STRUCTRAN-1 System Design and Implementation Manual.

5. TYPE OF REPORT & PERIOD COVERED
Final Technical Report.
May 75 – Jan 76

6. PERFORMING ORG. REPORT NUMBER
N/A

10 7. AUTHOR(s)
D. M./Andrews
R. A./Melton

15 8. CONTRACT OR GRANT NUMBER(s)
F30602-75-C-0245

9. PERFORMING ORGANIZATION NAME AND ADDRESS
General Research Corporation
P. O. Box 3587
Santa Barbara CA 93105

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
32010314

11. CONTROLLING OFFICE NAME AND ADDRESS
Rome Air Development Center (ISIS)
Griffiss AFB NY 13441

11 12. REPORT DATE
August 1976

13. NUMBER OF PAGES
14

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
Same

12 18p.

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
N/A

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited.

16 AF-32Ø1

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
Same

17 32Ø1Ø3

18. SUPPLEMENTARY NOTES
RADC Project Engineer:
Donald L. Mark (ISIS)

DMAAC Project Engineer:
Ms. Opal Power

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Structured Programming
Precompilers
Translators
Software Tools

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The STRUCTRAN-1 System Design and Implementation Manual provides a high-level description of the processing performed by STRUCTRAN-1, defines STRUCTRAN-1 file usage, and documents the use of common variables by STRUCTRAN-1. Appendix A describes the function of STRUCTRAN-1 subroutines in detail.

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

402 754
LB

CONTENTS

1 INTRODUCTION

The STRUCTRAN structured programming translators are tools for enabling structured programming in FORTRAN-based software developments. An extension of FORTRAN called DMATRAN replaces FORTRAN control statements with five statement constructs that lend themselves to GOTO-free structured programming. STRUCTRAN-1 translates programs written in DMATRAN into pure FORTRAN programs. STRUCTRAN-2 translates programs written in FORTRAN into a structured form expressed in DMATRAN.

The STRUCTRAN-1 software has been developed on the Control Data Corporation (CDC) 6400 computer at the General Research Corporation (GRC) facility in Santa Barbara, California. Following this software development phase, STRUCTRAN-1 has been installed on the UNIVAC 1108 at the Defense Mapping Agency Aerospace Center (DMAAC), St. Louis, Missouri. The software development phase utilized the CDC FORTRAN run compiler, version 2.3, under the GOLETA operating system for the CDC 6400. The installation phase utilized the UNIVAC 1108 FORTRAN V compiler with the UNIVAC 1108 operating system.

This document describes the overall design of the STRUCTRAN-1 software (Sec. 2) and the organization and contents of the STRUCTRAN-1 data base (Sec. 3). A narrative description of each module of STRUCTRAN-1 is included in Appendix A. The details of the STRUCTRAN-1 organization, intermodule dependencies, and intramodule control structure are contained in the STRUCTRAN-1 Software Analysis Collection documentation.

2 STRUCTRAN-1 PROGRAM OVERVIEW

2.1 STRUCTRAN-1 PROGRAM ORGANIZATION

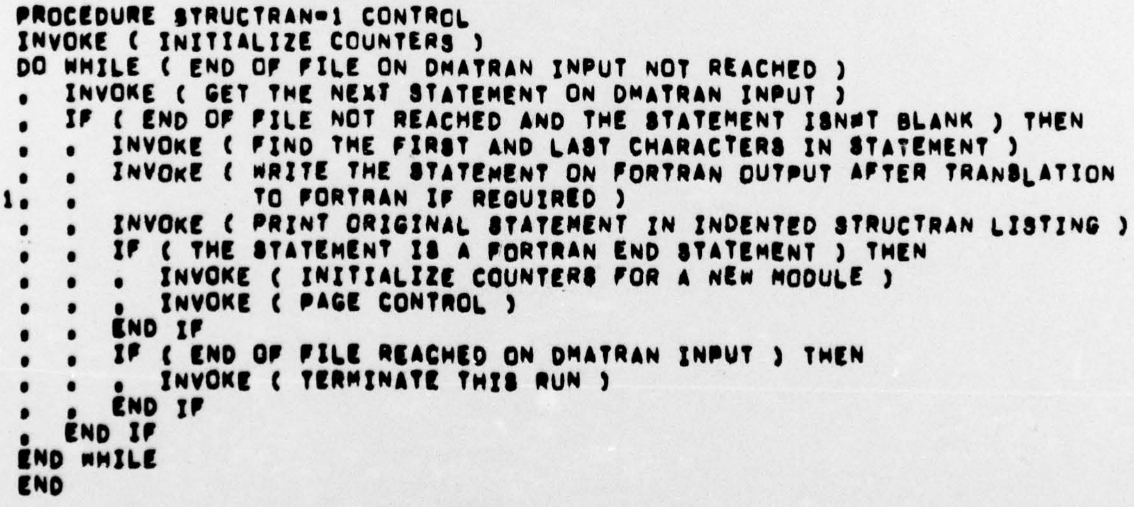
STRUCTRAN-1 is organized as a hierarchy of 50 subroutines and functions, each of which has a specific processing capability. STRUCTRAN-1 program operation is described in Sec. 2.2, and input/output file usage is spelled out in Sec. 2.3.

The code for all STRUCTRAN-1 modules was developed by GRC personnel utilizing structured programming techniques throughout. STRUCTRAN-1 was also implemented so as to avoid as many machine dependencies as possible. The equivalent FORTRAN version of STRUCTRAN-1 was used to install STRUCTRAN-1 on the UNIVAC 1108 at the Defense Mapping Agency Aerospace Center (DMAAC), in St. Louis, Missouri.

2.2 STRUCTRAN-1 OPERATION

STRUCTRAN-1 performs the translation from DMATRAN to FORTRAN on a statement-by-statement basis during a single pass through the DMATRAN source code. To avoid any word size dependencies, all character strings are internally represented in an A1 format (one character per word). Due to the simplicity of the structured syntax, STRUCTRAN-1 has to recognize only the various structured statement forms and the FORTRAN END statement to perform the translation process.

Figure 2.1 describes the sequence of operations performed by STRUCTRAN-1. The initialization function is performed at the start of a STRUCTRAN-1 run.



```
 PROCEDURE STRUCTRAN-1 CONTROL
 INVOKE ( INITIALIZE COUNTERS )
 DO WHILE ( END OF FILE ON DMATRAN INPUT NOT REACHED )
 .  INVOKE ( GET THE NEXT STATEMENT ON DMATRAN INPUT )
 .  IF ( END OF FILE NOT REACHED AND THE STATEMENT ISN#T BLANK ) THEN
 .  .  INVOKE ( FIND THE FIRST AND LAST CHARACTERS IN STATEMENT )
 .  .  INVOKE ( WRITE THE STATEMENT ON FORTRAN OUTPUT AFTER TRANSLATION
1.  .              TO FORTRAN IF REQUIRED )
 .  .  INVOKE ( PRINT ORIGINAL STATEMENT IN INDENTED STRUCTRAN LISTING )
 .  .  IF ( THE STATEMENT IS A FORTRAN END STATEMENT ) THEN
 .  .  .  INVOKE ( INITIALIZE COUNTERS FOR A NEW MODULE )
 .  .  .  INVOKE ( PAGE CONTROL )
 .  .  END IF
 .  .  IF ( END OF FILE REACHED ON DMATRAN INPUT ) THEN
 .  .  .  INVOKE ( TERMINATE THIS RUN )
 .  .  END IF
 .  END IF
 END WHILE
 END
```

Figure 2.1 STRUCTRAN-1 Control

This includes modification of any of the default parameters (as described in the STRUCTRAN-1 User's Manual). The following operations are performed on a statement-by-statement basis until an end-of-file is encountered on the DMATRAN input unit. The next complete statement is input; this may consist of up to 20 card images. If the statement is not blank, its first and last characters are identified. All statements are examined to see if translation to FORTRAN is required. A DMATRAN statement translates into one or more FORTRAN statements (the Final Report contains a complete description of the translation templates). Each original FORTRAN and translated FORTRAN statement is written to the FORTRAN output unit. Also each original statement is included in the indented listing produced by STRUCTRAN-1. If the statement is a FORTRAN END statement then the counters are reinitialized for a new module and page control skips to a new page. If the end-of-file on DMATRAN input has been reached the STRUCTRAN-1 run is terminated.

2.3 STRUCTRAN-1 FILE USAGE

In performing its translation function, STRUCTRAN-1 utilizes the files shown in Table 2.1. This is also summarized in Fig. 2.2. The DMATRAN source to be translated to FORTRAN is read in on unit LUNIN. STRUCTRAN-1 writes an indented listing on unit LUNOUT, and prints diagnostics about invalid constructs on LUNERR. These two unit names should normally refer to the same physical unit. The translated FORTRAN source is written onto unit LUNFOR. An example of each input and output format can be found in the STRUCTRAN-1 User's Manual.

TABLE 2.1

FILES USED IN STRUCTRAN-1 PROCESSING

File Number	File Name	Data Structure	Mode	Storage Format	Record Format	Recommended Allocation
5	LUNIN	DMATRAN input	BCD	Sequential	Card image	System card reader
6	LUNOUT	Reports	BCD	Sequential	Maximum 132 characters per line	System printer
2	LUNFOR	FORTRAN output	BCD	Sequential	Card image	Scratch file
6	LUNERR	Reports	BCD	Sequential	Maximum 132 characters per line	System printer

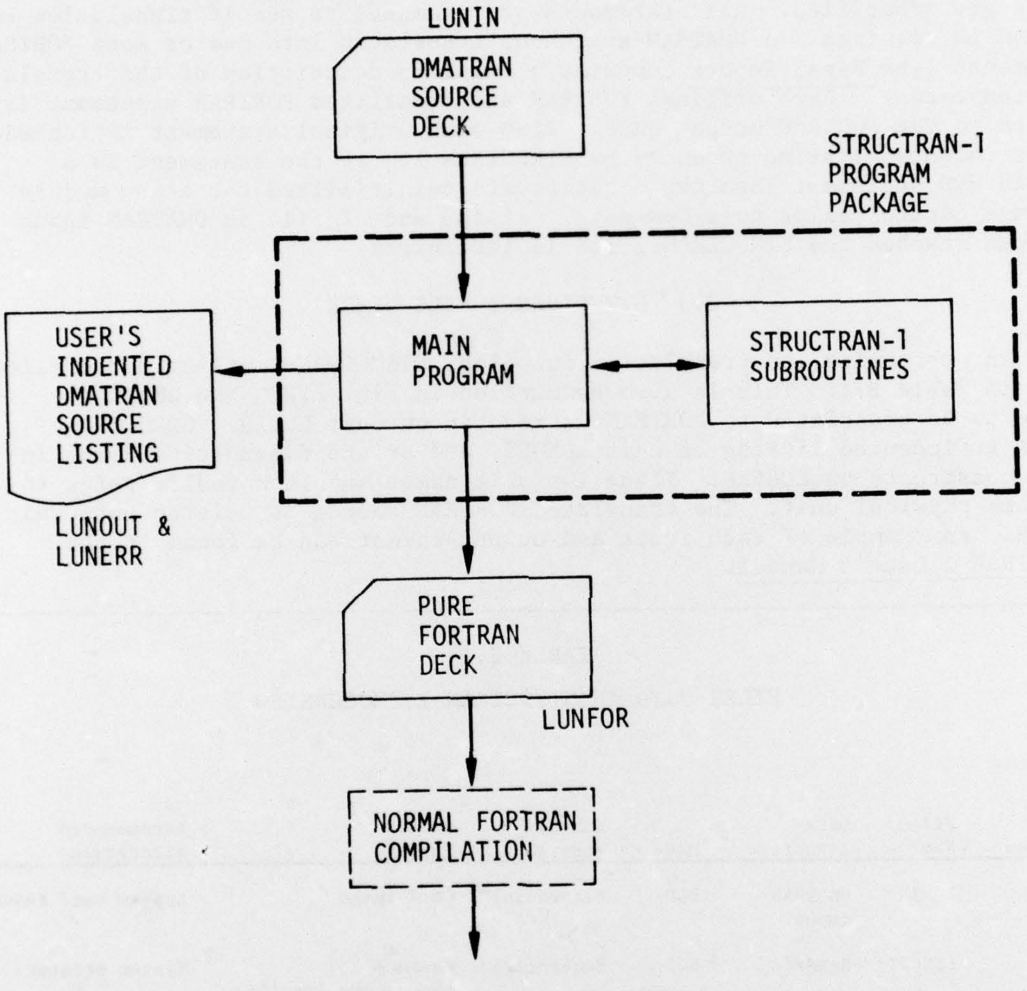


Figure 2.2. STRUCTRAN-1 File Activity

3 STRUCTRAN-1 VARIABLE DESCRIPTIONS

The following information indicates the common blocks which are used in STRUCTRAN-1. For each common block, a description of the use of all variables contained in it is included. For more information concerning STRUCTRAN-1 common blocks and variables, refer to the STRUCTRAN-1 Software Analysis Collection documentation.

```
COMMON/USEOPT/LUNIN,LUNOUT,LUNFOR,LUNERR,
1       INDON,NINDNT,INDCOM,LADJST,LINPER,
1       NLBLS,LINWDT,KOMFTN

THIS IS THE USER OPTIONS COMMON.

LUNIN  = LOGICAL UNIT FOR INPUT
LUNOUT = LOGICAL UNIT FOR DMATRAN OUTPUT.
LUNFOR = LOGICAL UNIT FOR FORTRAN OUTPUT
LUNERR = LOGICAL UNIT FOR ERROR OUTPUT
INDON  = 1 IF INDENTING REQUIRED, 0 IF NOT
NINDNT = NUMBER OF SPACES TO INDENT AT EACH LEVEL
INDCOM = 1 IF COMMENTS ARE TO BE INDENTED, 0 IF NOT.
LADJST = 1 IF STATEMENT TEXT IS TO BE LEFT ADJUSTED, 0 IF NOT.
LINPER = NUMBER OF DMATRAN LINES PER PAGE
NLBLS  = NUMBER AT WHICH TO START MAKING DMATRAN LABLES
LINWDT = DMATRAN OUTPUT LINE WIDTH
KOMFTN = 1 IF COMMENTS ARE TO BE PASSED TO FORTRAN OUTPUT, 0 IF
         NOT.
```

```
COMMON/CARDS/ICARD(80),IEOF,NUMCRD

THIS IS THE INPUT SPACE COMMON

ICARD  = CARD IMAGE ARRAY IN A1 FORMAT
IEOF   - 1 IF END OF DATA WAS SENSED, 0 IF NOT.
NUMCRD = INPUT CARD NUMBER
```

```
COMMON/STATE/LABEL(6),LIST(1420),ISEQ(8,20),NSTATE
1 ,LINBEG,LINEND,LENGTH,MENGTH,LPOINT,LEVEL,ITYPE,LTYPE

THIS IS THE STATEMENT COMMON

LABEL  = ARRAY OF 6 CHARACTER LABELS (IN A1 FORMAT) FOR A
         STATEMENT WITH UP TO 19 CONTINUES.
LIST   = ARRAY OF A1 CHARACTERS FOR A 20-CARD STATEMENT TEXT
         PLUS RESERVE SPACE FOR MAXIMUM INDENTATION DEPTH OF 100
         CHARACTERS
ISEQ   = ARRAY OF 8-CHARACTER SEQUENCE NUMBERS FOR A 20-CARD
         STATEMENT.
NSTATE = STATEMENT NUMBER
LINBEG = CHARACTER NUMBER WITHIN A STATEMENT FOR THE FIRST NON-
         BLANK CHARACTER OF THE STATEMENT.
LINEND = CHARACTER NUMBER WITHIN A STATEMENT FOR THE LAST NON-
         BLANK CHARACTER OF THE STATEMENT.
LENGTH = LINEND-LINBEG+1
MENGTH = LENGTH OF TEXT EXCLUSIVE OF DMATRAN COMMAND, IE TEXT
         FOLLOWING  <IF(> IN AN IF STATEMENT
LPOINT = POINTS TO FIRST TEXT CHARACTER AFTER DMATRAN COMMAND
         (SEE DEFINITION OF -MENGTH-)

LEVEL  = INDENTATION LEVEL
ITYPE  = DMATRAN STATEMENT TYPE
LTYPE  = TYPE OF PREVIOUS DMATRAN STATEMENT
```

```
COMMON/INTERN/NLINES,LINWID,KSTMT,NFATER,NEWRTN,NCOMAS,KOMMAS(50)

THIS IS THE INTERNAL VARIABLES COMMON.

NLINES = LINE NUMBER FOR PAGING
LINWID = LINE WIDTH FOR TEXT DMATRAN OUTPUT ( LESS CARD NUMBER ,
         LABEL AND SEQUENCE)
KSTMT  = NUMBER OF LINES READ IN CURRENT STATEMENT
NFATER = NUMBER OF FATAL ERRORS ENCOUNTERED.
NEWRTN = 1 WHEN LABLING SHOULD BEGIN AT NLBLS
NCOMAS = NUMBER OF COMMAS AT 1 LEVEL OF PARENTHESIS NESTING.
KOMMAS = ARRAY OF 1-LEVEL COMMA LOCATIONS IN ARRAY OF INPUT DATA.
```

```
COMMON/FORTRN/KABEL(6),KENGTH,KFTN(1420),KFLAG

THIS IS THE FORTRAN OUTPUT COMMON

KABEL  = LABEL OF FORTRAN STATEMENT
KENGTH = LENGTH OF FORTRAN STATEMENT
KFTN   = ARRAY OF FORTRAN CHARACTERS TO PLACE ON FORTRAN OUTPUT
         UNIT.
KFLAG = LEFT JUSTIFICATION FLAG.
```

```
COMMON/ACCTNG/NIFTRN,NFORT,NCOMMT,NTOTIF,NTOTFR,NTOTCM,NMODUL

THIS IS THE ACCOUNTING COMMON

NIFTRN = NUMBER OF DMATRAN STATEMENTS PROCESSED IN CURRENT MODULE
NFORT  = NUMBER OF FORTRAN STATEMENTS PROCESSED IN CURRENT MODULE.
NCOMMT = NUMBER OF COMMENT STATEMENTS PROCESSED IN CURRENT MODULE.
NTOTIF = TOTAL NUMBER OF DMATRAN STATEMENTS SO FAR.
NTOTFR = TOTAL NUMBER OF FORTRAN STATEMENTS SO FAR.
NTOTCM = TOTAL NUMBER OF COMMENT STATEMENTS SO FAR.
NMODUL = TOTAL NUMBER OF MODULES PROCESSED.
```

```
COMMON/CONSTN/LBK

THIS IS THE COMMON FOR CONSTANTS.

LBK     = HOLLERITH BLANK
```

```
COMMON/STACK/MAXSTK,INSTAK,LSTACK(4,50)

THIS IS THE STACKING COMMON

MAXSTK = MAXIMUM NUMBER OF ENTRIES IN THE STACK
INSTAK = POINTER TO CURRENT POSITION IN STACK
LSTACK(1,-) = STATEMENT TYPE
LSTACK(2,-) = LABEL1(POSITIVE INTEGER)
LSTACK(3,-) = LABEL2(POSITIVE INTEGER)
LSTACK(4,-) = LABEL3(POSITIVE INTEGER)
```

```
COMMON/RECNIZ/IRECLG(22),IREC(15,22)

THIS IS THE RECOGNITION COMMON

IRECLG = LENGTH OF NON-BLANK KEY PART OF STATEMENT
IREC   = ARRAY OF KEY WORDS AND KEY STATEMENT BEGINNINGS
```

```
COMMON/STYPE/INDTYP(22),IGRP(22),IPARCK(22)

THIS IS THE STATEMENT TYPE COMMON

INDTYP = +1 FOR INDENT NEXT STATEMENT, -1 FOR BRING IN THIS
         STATEMENT, 0 FOR LEAVE ALONE.
IGRP   = STATEMENT GROUP FOR ERROR CHECKING
IPARCK = 1 FOR CHECK PARENTHESES, 0 OTHERWISE.
```

```
COMMON/ESE/NENTRY,NRTRN

THIS COMMON SAVES THE NUMBER OF ENTRY POINTS AND RETURNS

NENTRY = NUMBER OF ENTRY POINTS
NRTRN  = NUMBER OF RETURNS
```

```
COMMON/TRACE/NALTER

THIS IS THE COMMON TO PRINT OUT STATEMENT NUMBERS

NALTER = CONTROLS OUTPUT OF STATEMENT NUMBERS WHICH REFER
         BACK TO THE DMATRAN CODE.
```

```
COMMON/WARNIN/IWARN

THIS IS THE COMMON TO KEEP TRACK OF UNSTRUCTURED FORTRAN STATEMENT

IWARN  = FLAG SET TO 1 WHEN UNSTRUCTURED FORTRAN STATEMENT IS
         ENCOUNTERED, OTHERWISE EQUALS 0
```

```
COMMON/INVOKE/NOBE,NOBLOK(20,6),NAME1(6),NOINV(20)

THIS IS THE INVOKE COMMON

NOBE   = NUMBER OF UNIQUE BLOCK NAMES ENCOUNTERED IN INVOKE
         STATEMENTS IN THE CURRENT MODULE.
NOBLOK = LIST OF 6-CHARACTER BLOCK NAMES (IN ORDER OF ENCOUNTER)
         THAT ARE IN THE CURRENT MODULE
NAME1  = 6 CHARACTER WORK AREA FOR GENERAL PURPOSE USE.
NOINV  = THE NUMBER OF INVOKE STATEMENTS FOR EACH BLOCK
         IN THE SAME ORDER AS IN NOBLOK.
```

APPENDIX A

STRUCTRAN-1 MODULE FUNCTION DESCRIPTIONS

ACT1 is an accounting routine which is called at the end of the processing of each statement. It adds to the cumulative totals of either COMMENT, DMATRAN, FORTRAN, ENTRY or RETURN statements for the current module.

ACT2 is a second accounting routine. At the end of a module, the number of COMMENT, DMATRAN, and FORTRAN statements accumulated by ACT1 accounting is printed and then added to the totals so far for each type. If the number of ENTRY or RETURNs is greater than 1, then those totals are also printed.

ACT3(IBEG) is the third accounting routine. If IBEG = 0 , the variables in the accounting common are initialized to zero at the beginning of a STRUCTRAN-1 run. At the end of a run, IBEG is set to 1 and the total number of statements is printed out in addition to the DMATRAN, FORTRAN, and COMMENT accumulated totals.

ASSIGN forms the statement, <NI><IVAR> = <TEXT> in the common block FORTRN where <NI> = (LABEL(K),K=1,5) , <IVAR> = (IVAR(K),K=1,6) and <TEXT> = (LIST(K),K=1,L) where L = 1420 maximum.

BGSCAN sets the common variable LINBEG to the number of the first non-blank character of a statement.

CONT forms the statement, <NI> CONTINUE, in the output FORTRAN area. <NI> is equivalent to (LABEL(K),K=1,5).

CONTRL is the principal STRUCTRAN-1 control routine. It directs the processing by statement and by module. When an end-of-file is reached, it calls ENDER to terminate the run. CONTRL is called from the MAIN program.

CPTIME(TIME) belongs to the implementation dependent section. At implementation/installation time, installer must provide a call to a subroutine which produces the elapsed CP time in the variable TIME. Alternatively, TIME may be set to 0.0 if internal performance measurement data printed at the end of the STRUCTRAN-1 run is to be ignored. The output parameter, TIME, is floating point and is in seconds.

ENDER belongs to the implementation dependent section. It terminates the run. At implementation/installation time, installer must provide for consequences of termination of operation if that termination is to be other than to stop. The variable NFATER contains the number of fatal errors, if any, found in error processing.

ERROR(N) prints an error message. The output parameter, N, is the error number. The error messages are listed in Table 3.1 in the User's Manual.

FILCHK(LUNIT) is a function which checks for either physical end-of-file or END in columns 1 through 3. If either is found, the returned function value is 1; otherwise, 0.

FULCON(LABEL) calls CONT to create a continue statement whose label is LABEL, if LABEL(I),I=1,5 is non-blank.

GENASS(ILABEL,IVAR) forms a statement according to the following example: ASSIGN 99995 to I99996, where the array ILABEL contains the character string 99995 and the array IVAR contains the character string I99996.

GENGO(INDEX,IRVAR,INVLAB) is a subroutine to generate a statement of the form: GO TO <RETURN VAR> (<LIST OF INVOKE LABELS>) in the common block FORTRN. The <LIST OF INVOKE LABELS> contains the label of the first statement following each invocation of a particular BLOCK. The value of the <RETURN VAR> indicates which of these labels in the list to go to. For example, in GO TO I99969, (99994,99634,99429) if I99969 contained the value 99634 then transfer would be to the statement with that label. This would be the first statement which followed that INVOKE of the BLOCK.

GENLAB(N,LTARG) converts the positive integer N into character string format to be used as a label for the additional FORTRAN statements which are created by STRUCTRAN-1. N is the integer label and LTARG is the character label which is the output parameter.

GENVAR(N,IVAR) generates a 6-character integer variable name which starts with I followed by positive integer N; e.g., I99968. The variable name is stored in (IVAR(I),I=1,6).

GETCRD is a subroutine which reads one card and checks for end of data.

GETINS belongs to the implementation dependent section. At implementation/installation time, installer can provide code to set options away from default values pre-supplied here. This segment of code must be replaced if the system implementor does not want the default options listed below. The twelve basic options are read, at this point, from the first input card from unit 5. The format of this card is 12I5. Default values are supplied for each field which is blank. The options are read in the following order:

		Default
1.	STRUCTRAN-1 input unit (DMATRAN source)	5
2.	STRUCTRAN-1 listing unit (indented listing)	6
3.	Translated FORTRAN unit (to be compiled)	2
4.	STRUCTRAN-1 error diagnostics	6
5.	Whether to automatically indent (0 = NO; 1 = YES)	1
6.	The number of characters per indent level	3
7.	Whether to indent comments (0 = NO; 1 = YES)	0
8.	Whether to left-adjust all statements first (0 = NO; 1 = YES)	1

		Default
9.	Number of lines per page	57
10.	Number of initial generated label	99998
11.	Number of characters per line	132
12.	Whether to move comments to the FORTRAN file (0 = NO; 1 = YES)	0

GETSTM is a routine which brings in one statement at a time. If the statement is FORTRAN or DMATRAN, ITYPE is set to 1; if it is a COMMENT, ITYPE becomes 0.

GOTO(LTARG) forms the statement, GOTO <TARGET>, in the common block FORTRN. <TARGET> = (LTARG(K),K=1,5).

HOLLER(I) is called from IBALPR when an "H" is encountered in the array LIST(I). It determines whether the "H" indicates the presence of a Hollerith string or if it is a character of a variable name. If a Hollerith field is found, then I is incremented by the length of that field so that if a parenthesis is contained within the field, IBALPR will not assume it is the parenthesis for which it is searching.

IBALPR is a function which examines the common vector LIST starting at LPOINT and returns the character position of the first balancing closing parenthesis; if none is found before LINEND, 0 is returned.

IFCASE(LAB,IVAR,MSTRNG,LIST,LTARGT,LPOINT) forms the statement <NI> IF (<VAR>.NE.(<TEXT1>).AND.<VAR>.NE.(<TEXT2>).AND.... <VAR>.NE.(<TEXTN>)) GO TO <TARGET> in the FORTRAN output area where:

```
<NI>     = (LAB(I),I=1,5)
<VAR>    = (IVAR(I),I=1,6)
<TEXT1>  = (LIST(I),I=1,N1) where LIST(N1+1) = 1H, or end of text
<TEXT2>  = (LIST(I),I=N1+2,N2) similarly
   :       no more than 50 of these
<TARGET> = (LTARGT(I),I=1,5)
```

LPOINT is the character position of the start of LIST on the calling program.

IFEOF(LUN) belongs to the implementation dependent section. At implementation/installation time, installer must provide logically correct end-of-file check operations if the code herein included is incompatible with that implemented on his machine. This subroutine is called once for each statement, and should return the value 0 unless an EOF is found. Alternatively, STRUCTRAN-1 makes a separate internal check of the first three columns of each input card. STRUCTRAN-1 operations cease if those columns contain END. This function should return a value of 1 when EOF or END is found on LUN. Otherwise, a 0 is returned. Typical call:

```
IF(IFEOF(LUN).EQ.1) THEN
```

The input parameter, LUN, is the logical unit to be tested. If DMATRAN source on LUN is terminated by end file, this function must be adapted to operate as described above. For example, the following code operates on CDC machines:

```
         IF (EOF,LUN) 10,20
   10    IFEOF = 1
         RETURN
   20    IFEOF = 0
         RETURN
         END
```

If DMATRAN source on LUN is terminated with a card that contains the word END starting in column 1, this routine is a dummy but must be included.

IFSO(LABEL,L,LIST,LTARG) forms the statement <NI> IF (<TEXT>) GO TO <TARGET> in the common block FORTRN:

<NI> = (LABEL(K),K=1,5)

<TEXT> = (LIST(K),K=1,L) including 1 closing parenthesis

<TARGET> = (LTARG(K),K=1,5)

IGROUP(X) returns 0 for stack-error free, 1 for stack error, and 2 for empty stack. It checks for top stack position in same group.

INDENT(NIN) indents a statement prior to STRUCTRAN-1 output.

INDLEV computes indentation level.

INITAL is called at the beginning of a STRUCTRAN-1 run to initialize common data.

ISEND is a function which is called by GETSTM to determine if the current statement is an END statement, indicating termination of the current module.

IWITHN(X) function returns 0 for error-free, 1 for within group error, and 2 for empty stack.

KEMPTY(LABEL) is a function that returns 1 if (LABEL(I),I=1,5) is all blank; otherwise, it returns a 0.

KLASS classifies a non-comment statement according to type and, if necessary, resets the common variable ITYPE. The following is a list of the various classifications:

ITYPE	DATA RECOGNIZED	ITYPE	DATA RECOGNIZED
2	IF	11	DO UNTIL
3	ELSE	12	END UNTIL
4	END IF	13	THEN
5	DO WHILE	14	GOTO
6	END WHILE	15	INVOKE
7	CASE OF (	16	BLOCK
8	CASE (	17	END BLOCK
9	CASE ELSE	18	RETURN
10	END CASE	19	ENTRY

KLASS1 first calls KLASS to set the ITYPE. KLASS1 also allows the IF( ) THEN structured syntax to be recognized, as well as the IF( )GOTO statement in standard FORTRAN. If it is the latter, the flag, IWARN, is set to 1 so a warning message will be printed.

KCOMP(I,J) sets its function value to 1 if BCD character I is equal to BCD character J; if not, it is set to 0. Typical call is:

```
        IF (KCOMP(L,M) .EQ. 1) THEN
```

The input parameters I and J are the characters to compare.

LADJUS left-adjusts the text of a statement.

MAIN is the driver program which calls CONTRL to initiate the processing of a STRUCTRAN-1 run.

MOVEWD(NWDS,IPOS,IARRY,1POS1,IARRY1) is a data transfer routine which copies from IARRY(IPOS) to IARRY1(IPOS1). The input parameters are:

NWDS = number of words to be moved

IPOS = index in IARRY at which to start copying

IARRY = array containing words to be copied

IPOS1 = index to IARRY1 at which to start putting copied data

IARRY1 = array to which data is copied

When IARRY does not equal IARRY1, either MOVEWD or MOVEWU may be used. However, when data is being copied from one position to another in the very same array, then it is important to note the differences in these two routines.

When IPOS is greater than IPOS1, then MOVEWD is the appropriate routine to use. For example, a typical move might be CALL MOVEWD(27,14,X,3,X), which would move 27 words beginning with X(14) to X(3).

MOVEWU(NWDS,IPOS,IARRY,IPOS1,IARRY1) is the correct routine to call when IPOS1 is greater than IPOS and IARRY is equal to IARRY1, as in CALL MOVEWU(27,3,X,14,X). The reason for this distinction between the two similar routines, MOVEWD and MOVEWU, is to prevent overwriting of a part of an array and the consequent loss of data before that data can be copied to its new location in the array.

NAMOB(X) is a function which searches the array NOBLOK for a match with the block name residing in <LIST> starting at <LPOINT>. It returns the index in the array where found or 0 if not found.

NDSCAN sets the common variable LINEND to the number of the last non-blank character of a statement.

NEWLAB(X) is a function which generates a positive integer label by counting downward initially from the label which was stored in NLBLS in subroutine GETINS. Unless the default parameter is changed, NLBLS is first 99998; and each time NEWLAB is called, 1 is subtracted from it.

NEWPAG controls paging on STRUCTRAN-1 output before writing one line.

PUTFTN(KOMT) writes a statement on FORTRAN output, making each line a comment if KOMT is 1.

PUTIF causes printing of a whole statement on STRUCTRAN-1 output by calling PUTIFT after first checking if it is necessary to left-adjust the text.

PUTIFT prints a statement on STRUCTRAN-1 output unit.

SPRYWD(NWDS,IVALUE,IPOS,IARRY) sprays one data value into the designated part of an array. A typical call is:

```
CALL SPRYWD(27,1H ,3,X)
```

It would blank 27 cells starting at X(3). The input parameters are:

NWDS = number of words to be filled with IVALUE.

IVALUE = the value to be placed

IPOS = index in IARRY at which to start putting value

The output parameter is:

IARRY = the array to be filled

STRUCT is the main routine which translates DMATRAN to the FORTRAN source code which is accepted by the FORTRAN compiler.

If a statement is not a comment, then KLASS1 is called which in turn calls KLASS to reset the variable, ITYPE, if necessary, thereby delineating various words, most of which are structured forms. [A list of the ITYPEs and the corresponding classifications may be found in the description of the subroutine KLASS.]

When ITYPE is between 2 and 17, the current statement is moved to the FORTRAN output unit and printed as a comment to improve understanding of the code which is input to the FORTRAN compiler. Next the appropriate action is implemented according to the ITYPE that has been delineated.

For example, if the statement is a DO WHILE (ITYPE = 5), then the LSTACK pointer, INSTAK, is incremented; three new labels are generated and stored in LSTACK at positions 2, 3, and 4; and ITYPE is entered at LSTACK(1,INSTAK). The necessary source code is generated with the labels (accessible in LSTACK) inserted in the proper places in the FORTRAN statement.

If there is another DMATRAN control type within the DO WHILE, INSTAK will be incremented again and new labels stored in the newly available locations. These will be used and then INSTAK will be decremented so that when the END WHILE (ITYPE = 6) is reached, the stack pointer will be accessing the labels which were generated for the DO WHILE. Then the GO TO and CONTINUE statements, required for the translation into FORTRAN at the end of a DO WHILE iteration, will have the correct labels.

Should an END IF, END CASE, END UNTIL, END WHILE, or END BLOCK be missing, then an error will be detected and a reference to an error message will be pointed out.

VERBAT moves a statement directly into FORTRAN output area verbatim.

WARN is called by PUTIFT when the flag, IWARN, has been set to 1. It then transfers the warning message, UNSTRUCTURED FORTRAN STATEMENT, into LIST so it may be printed out on the STRUCTRAN-1 output unit.

IWARN is set to 1 in subroutine STRUCT when it finds a FORTRAN GOTO statement and in KLASS1 when an IF( ) GOTO construct has been established. These two forms, being correct standard FORTRAN code, will be accepted and processed correctly by the compiler. However, for uniformity, they should be changed to the DMATRAN equivalent.